ARMY RESEARCH LABORATORY

Object Characterization in Greyscale Imagery Using Fractal Dimension

by Philip Beaver and Stephanie M. Quirk
(US Military Academy)
Joseph P. Sattler
(Army Research Laboratory

ARL-TR-749

September 1995

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE September 1995	3. REPORT TYPE AND DATES COVERED Final, June 1993 to December 1994

4. TITLE AND SUBTITLE Object Characterization in Greyscale Imagery Using Fractal Dimension	5. FUNDING NUMBERS DA PR: AH44 PE: 61102
6. AUTHOR(S) Philip Beaver, Stephanie M. Quirk (US Military Academy), and Joseph P. Sattler (Army Research Laboratory)	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory Attn: AMSRL-SE 2800 Powder Mill Road Adelphi, MD 20783-1197	8. PERFORMING ORGANIZATION REPORT NUMBER ARL-TR-749
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory 2800 Powder Mill Road Adelphi, MD 20783-1197	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

AMS code: 611102.H44
ARL PR: 5AE119

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Local fractal dimension operators can be used to discriminate among objects in greyscale imagery. The mass scaling (box-counting) algorithm for computing local fractal dimension promises real-time results when screening numerous or large images. In this paper we show the ambiguities in the mass scaling technique based on shape and range, we examine previously published algorithms for computing local fractal dimension, we show by example some of their deficiencies, and we propose a new algorithm that solves some of these problems.

14. SUBJECT TERMS Greyscale imagery, object identification, fractal dimension	15. NUMBER OF PAGES 16
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

Contents

1. Introduction 5
2. The Fractal Dimension of Natural Objects 5
3. Local Fractal Dimension 6
4. Measure of "Roughness" 6
5. Scaling Pixel Values 7
6. Linear Regression Procedure 8
7. A Local Fractal Dimension Operator 10
8. Implementation and Modifications 12
References 14
Distribution 15

Figures

1. Greyscale values for three 5×5 image segments 7
2. Image segments in figure 1 scaled to interval [–10, 10] 7
3. Plots of the best-fit lines for the smooth and rough surfaces 9
4. Two segments of scaled image windows 11
5. Box-covering of scaled image windows 11
6. Modified box-counting by "stacking" boxes 12

Table

1. Box counting for scaled pixel values 9

1. Introduction

Ever since Benoit Mandelbrot proposed that natural objects may be better described with fractal geometry than with Euclidean geometry (Mandelbrot, 1983), there have been numerous attempts to characterize objects by their fractal dimension. An immediate application of this idea arises when we attempt to classify objects as subsets of greyscale images, whether the image was produced by radar, satellite imagery, x-ray, black-and-white photography, or some similar means. When such images are digitized, they are reduced to a range of greyscale values (often 0 through 255) that are assigned to a matrix of pixels. In this paper, we propose a method by which an object may be characterized by using a local fractal dimension operator on its digitized greyscale image.

The basis for this research comes from the requirement to identify man-made objects in images produced with synthetic aperture radar (SAR) mounted on an aerial platform, and specifically to screen images of interest, which can then be confirmed by more precise techniques. However, the procedure may also be applied to many other disciplines, particularly when numerous or large images must be screened and processing time is critical. We note that the Lincoln Laboratory uses the fractal dimension of objects in binarized SAR imagery to separate natural from manmade objects (Novak, 1993).

2. The Fractal Dimension of Natural Objects

The fractal dimension of a set A, as a subset of $\mathbf{R}^n$, is defined as

$$D = \lim_{\varepsilon \to 0} \frac{\ln\left[N(\varepsilon)\right]}{\ln\left(1/\varepsilon\right)}, \tag{1}$$

where $N(\varepsilon)$ is the minimum number of balls of diameter ε required to cover A. While there are alternative, and sometimes not equivalent (Hentschel, 1983), definitions of fractal dimension, most are derived from examining the limit of the ratio of the log of a set's "mass" to the log of the scale of observation, as the scale goes to zero. This definition is consistent with the concept of Euclidean dimension, and, in fact, is a generalization of that concept.

This mathematical definition presents a theoretical problem when we are determining the fractal dimension of a natural object. Objects in nature as subsets of $\mathbf{R}^3$ (or images as subjects of $\mathbf{R}^2$) will have differing fractal dimensions depending on the scale of observation. Since there are no true fractals in nature (Mandelbrot (1983) points out that there are no straight lines or perfect circles in nature either), we must modify the concept of fractal dimension if we are to apply it to natural objects.

This problem is solved by the selection of an appropriate scale, and for different classes of objects, the range of "ε-values" may vary widely (Turcotte, 1992). The appropriate scale for analyzing greyscale imagery is predetermined by the set of pixels over which the object is represented. For a greyscale image, ε can never be taken smaller than the size of a single pixel.

3. Local Fractal Dimension

As the purpose of our research is to classify and identify objects within an image, we need the concept of local fractal dimension. Instead of computing the fractal dimension of an image, we would like to compute the fractal dimension of subsets of the image, and use these local values to classify the subsets. Since our subset range has a lower bound of the size of one pixel, we assign a fractal dimension to each pixel, which forms the basis of our object classification.

Various techniques exist for computing a local fractal dimension, including mass scaling, power spectral scaling, variance scaling (Moghaddam, 1991), computing the Hurst coefficient (Gage, 1990), and lacunarity (Voss, 1986). While these methods contain common elements and all have a similar result (the assignment of a local fractal dimension to a single pixel based on a window of surrounding pixels), we are also interested in the speed of the algorithm for our application. This led us to select the mass scaling technique as the basis of our method, which is most similar to the "box-counting" technique of computing global fractal dimension. Voss (1986) proposed this type of local fractal dimension operator, and Keller and Chen (1988) published an algorithm to implement the procedure. While other studies found this to be the least accurate method (Moghaddam, 1990), it was the only one that promised real-time results, a feature critical to our application.

4. Measure of "Roughness"

Since a digitized image is essentially a discretization of a surface in three dimensions, we desire an operator that assigns a fractal dimension consistent with the "roughness" of the surface, i.e., higher fractal dimensions corresponding to rougher surfaces, with a value of two for a flat surface and a value of three for a space-filling surface.

The goal of having a higher fractal dimension assigned to a "rougher" surface leads to an ambiguity with local fractal dimension operators: While there are ways to measure surface roughness (Milman, 1994), what is lacking is a clear definition of what constitutes a "rough" surface in terms of mass scaling. We demonstrate this problem by example with three 5x5 "windows" of pixels about a central pixel, for which the local fractal dimension is to be computed.

0	200	0	200	0	90	110	90	110	90	50	0	100	150	200
200	0	200	0	200	110	90	110	90	110	50	0	100	150	200
0	200	100	200	0	90	110	100	110	90	50	0	100	150	200
200	0	200	0	200	110	90	110	90	110	50	0	100	150	200
0	200	0	200	0	90	110	90	110	90	50	0	100	150	200
		Case I					Case II					Case III		

Figure 1. Greyscale values for three 5×5 image segments.

Consider the three cases shown in figure 1. Case I is clearly more rough than case II, for although these two image segments have the same "shape" about the central pixel, the dynamic range is much greater in case I. Therefore, we would expect any local fractal-dimension operator to assign a higher value to the central pixel in case I.

Similarly, case I is rougher than case III. While both image segments have the same range of pixel values, the "surface" represented in case I fills much more space than that of case III, which is a relatively steep, albeit smooth, surface. The ambiguity arises when case II is compared with case III. While the "shape" of case II is much rougher, the "range" of case III is greater; since both the shape and the range of an image segment contribute to its roughness, it is impossible to find an operator that assigns a fractal dimension consistent with both criteria.

5. Scaling Pixel Values

We may try to avoid this ambiguity by assigning a fractal dimension based entirely on the shape of the image. To do this, we must scale all the pixels in the window to fall within a certain range, for example the interval [–10, 10], with the central pixel assigned a value of zero. The results of scaling the three cases in figure 1 are shown in figure 2.

Once the image segments have been scaled, the computed fractal dimension will be based entirely on the shape of the image, not on the range of pixel intensities, so cases I and II will have identical fractal dimensions. Similarly, as both scaled cases I and II are rougher than case III, they should have a higher fractal dimension than the smoother surface.

Since it may not be desirable for cases I and II to have the same fractal dimension (based on the greater pixel range of case I) once the fractal dimensions are computed for the scaled images, they could be rescaled by a weighting factor based on the range of the nonscaled window. This would

-10	10	-10	10	-10	-10	10	-10	10	-10	-5	-10	0	5	10
10	-10	10	-10	10	10	-10	10	-10	10	-5	-10	0	5	10
-10	10	0	10	-10	-10	10	0	10	-10	-5	-10	0	5	10
10	-10	10	-10	10	10	-10	10	-10	10	-5	-10	0	5	10
-10	10	-10	10	-10	-10	10	-10	10	-10	-5	-10	0	5	10
		Case I					Case II					Case III		

Figure 2. Image segments in figure 1 scaled to interval [–10, 10].

cause case I to again have a higher dimension than case II (as we would expect), but it may allow a smooth surface with a greater range to have a higher fractal dimension than a rough surface over a smaller range.

The decision to scale the pixel values or to rescale the computed fractal dimensions must be based on the image source itself, and some balance between these may be appropriate. For our application, where all pictures had approximately the same dynamic range, we found it was useful and consistent to scale every local window to the same range and to base the computed local fractal dimension entirely on the shape of the surface.

6. Linear Regression Procedure

In the mass scaling method of computing local fractal dimension, cubes of increasing size are considered about the central pixel, and for each size cube, the number of (scaled) pixels it contains are counted (Moghaddam, 1991, and Voss, 1986). For example, if we were to consider a 9×9 window about the central pixel, we would first scale all the pixel values so that the central pixel has a value of zero, and all others fall within the interval [–4.5, 4.5]. We could then consider "cubes" about the central pixel of sizes 1, 3, 5, 7, and 9, and we could count the number of scaled pixels contained in each cube. The cube of side 1 would necessarily contain the central pixel (since it was scaled to zero). The cube of side 3 would contain the central pixel and all pixels in the 3×3 grid about the central pixel with values in the interval [–1.5, 1.5]. Thus, for $n = 1, 3, 5, 7$, and 9, we could count $N(n)$, which is the number of pixels in the n×n grid about the central pixel that fall in the interval $[-n/2, n/2]$. Based on the scaling, $N(1) = 1$ and $N(9) = 81$, and $N(n)$ is a nondecreasing function.

The temptation at this point is to borrow from the technique used to compute global fractal dimension (Falconer, 1991) and take the natural logs of the n and $N(n)$ values, plot them against each other, use linear regression to find the best-fit line, and take the slope of that line as the local fractal dimension of the central pixel. Again, we use an example to show why the mass scaling method for computing local fractal dimension so often fails.

Consider two surfaces, one smooth and one as rough as possible. For the smooth surface, $N(1) = 1$, $N(3) = 9$, $N(5) = 25$, $N(7) = 49$, and $N(9) = 81$, in that every possible pixel is contained within every cube of increasing size. The most pathologically rough surface is one where no pixels (except the central one) are contained within any of the cubes of size less than nine, so that $N(1) = N(3) = N(5) = N(7) = 1$, and $N(9) = 81$. Table 1 shows a chart of the n and $N(n)$ values for these two surfaces, and figure 3 shows the regression lines for the natural logs of n and $N(n)$ for the smooth and rough surfaces, respectively.

For a smooth surface, the equation of the best-fit line for $\ln(n)$ and $\ln[N(n)]$ is $y = 2x$, and the assigned local fractal dimension is 2. However, for the rough surface, where we would expect a fractal dimension near 3, we find

Table 1. Box counting for scaled pixel values.

	$N(n)$	
n	Smooth surface	Rough surface
1	1	1
3	9	1
5	25	1
7	49	1
9	81	81

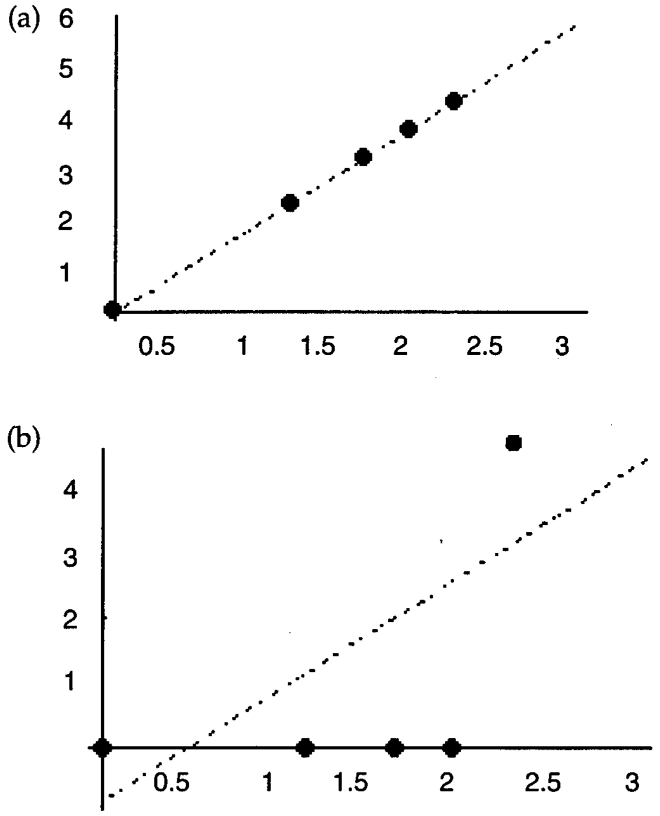


Figure 3. Plots of the best-fit lines for the smooth and rough surfaces. The abscissae are ln(n) and the ordinates are ln[$N(n)$].

the best-fit line is $y = 1.765x - 0.831$, and the assigned fractal dimension is 1.765. Clearly, counting boxes, taking logs, and performing linear regression does not suffice as a local fractal dimension operator, regardless of the speed of implementation.

To solve this problem with linear regression, it is possible to force the regression line through the point (ln(9), ln(81)), but while this will assign a higher fractal dimension for rougher surfaces, it sets a theoretical upper bound on the computed fractal dimension of 2.82 for a 9×9 window. (Increasing the window size increases this upper bound, possibly beyond three, and therefore this is an artificial "fix" that we found to be insufficient for our application.) Instead, we borrow a procedure from the box-counting method used to compute global fractal dimension, and propose this as a new method of computing local fractal dimension.

7. A Local Fractal Dimension Operator

In order to compute the local fractal dimension of a single pixel, a window of surrounding pixels must be selected. While the literature differs on the optimal window size, ranging from 9×9 (Moghaddam, 1991) to 41×41 (Keller, 1989), we must achieve a balance among getting enough information, having the procedure be computationally tenable, and getting so much information from a large window that the local characteristic of the operator is lost. This optimal balance is dependent on the image source, scale, and desired speed of processing, and must be determined for each application. For the algorithm we employ for our application, a 16×16 window about four central pixels was found to be appropriate.

Since we use a window of even size, we actually compute the local fractal dimension for a spot between four adjacent pixels; but since our target objects will have a size greater than one pixel, this does not create a problem with their identification. We begin by scaling to zero the average of our four central pixels, and linearly scaling all surrounding pixel values in the 16×16 window to the interval [–8, 8]. We then compute the local fractal dimension of the center point using a method similar to the "box-counting" technique for computing global fractal dimension, and different from the mass scaling method discussed in section 6.

The center of the 16×16 window, which has value zero, is the center of our domain (i.e., the plane containing the pixel positions). The range of scaled pixel values is [–8, 8], which corresponds to heights above or below the domain. This cube constitutes the space used for computing the local fractal dimension. We consider cubes of size 1, 1/2, 1/4, and 1/8, and count boxes to obtain $N(1)$, $N(1/2)$, etc., where $N(n)$ is the number of boxes in size n containing one or more scaled pixels.

Since a cube of size 1 covers the entire space (and thus all scaled pixel values), $N(1) = 1$. For this method, we next divide the space into eight octants and count each octant that contains a scaled pixel value to obtain $N(1/2)$. However, to proceed, we must modify the box-counting method for the reason shown in the following examples.

We demonstrate the modified box-counting technique with a two-dimensional example of determining $N(1/8)$. Consider the two image surface segments in figure 4, which have been scaled so that all pixel values fall in the interval [–8, 8]. This can be thought of as a slice of our space taken along one row of pixel positions in the domain. We have overlaid the segments with 8×8 grids, and have indicated the scaled pixels with points.

In figure 5 we cover these image surfaces with boxes, as is done in the traditional box-counting method; i.e., every box containing a pixel is counted to compute $N(1/8)$. Clearly, while the image in case I is more "rough,"

Figure 4. Two segments of scaled image windows.

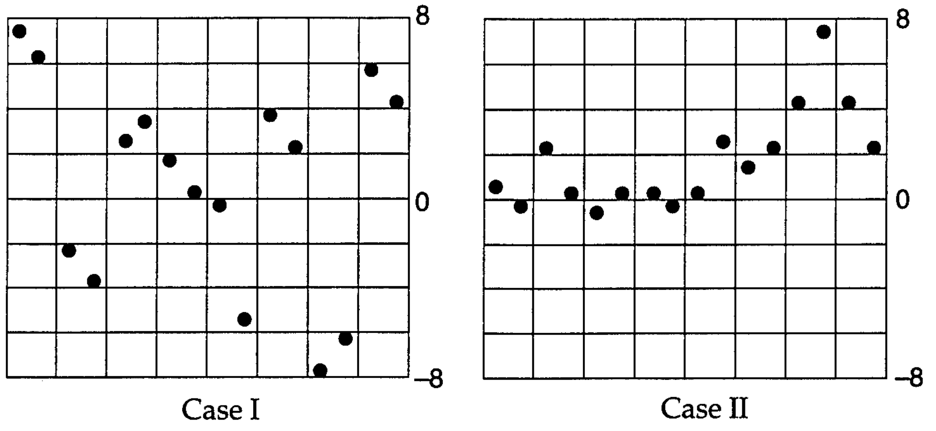
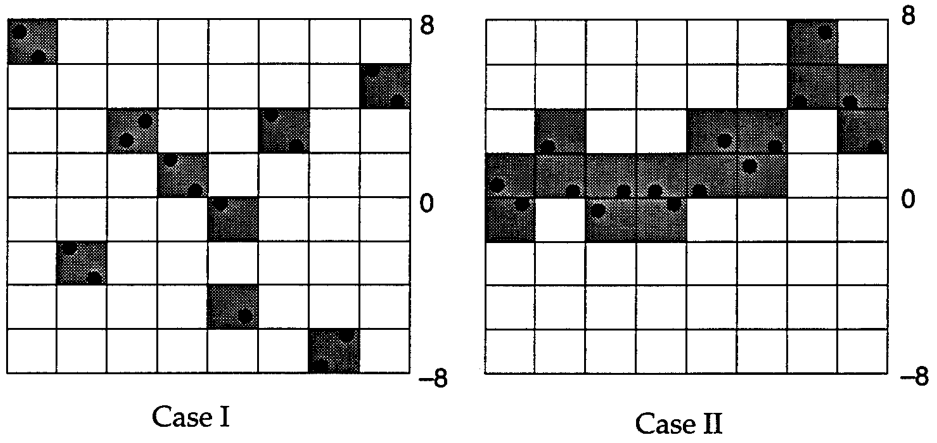


Figure 5. Box-covering of scaled image windows. $N(1/8) = 9$ for case I, while $N(1/8) = 16$ for case II.



$N(1/8)$ is greater for case II, which leads to a higher computed fractal dimension. Although the surface described by case I may fill more of the space, the discretization of the surface shows pixel values as points with no "fill" between them. Additionally, since we are computing the fractal dimension of a point that has been scaled to zero, pixel values of a greater magnitude should indicate a higher local fractal dimension.

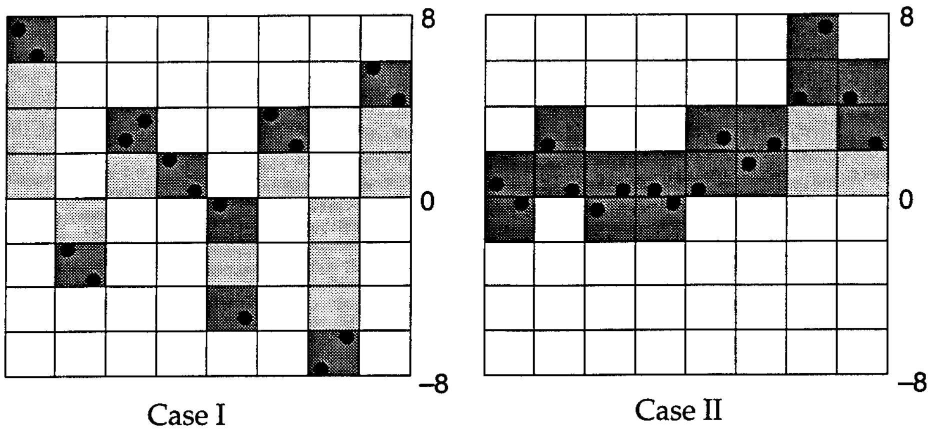
We therefore modify our box-counting technique as shown in figure 6, where we include in our count all the boxes between the pixels and the domain. This technique of "stacking" boxes from the plane of the domain not only measures the deviation from the central location, but it also measures the amount of space filled by the surface in the local window.

We use this modified box-counting technique ("stacking" boxes from the plane of the domain) to compute both $N(1/4)$ and $N(1/8)$.

Since we know the abscissae of the regression line (ln(1), ln(2), ln(4), and ln(8)), we express the fractal dimension slope (FD) of this line entirely in terms of $N(n)$ in the following manner:

$$\mathrm{FD} = \left[3\ln\left(N\left(\frac{1}{8}\right)\right) + \ln\left(N\left(\frac{1}{4}\right)\right) - \ln\left(N\left(\frac{1}{2}\right)\right)\right] \Big/ \left(10\ln(2)\right) .$$

Figure 6. Modified box-counting by "stacking" boxes. Now $N(1/8) = 21$ for case I, and $N(1/8) = 19$ for case II.



Additionally, $N(1/2) \in [4, 8]$, $N(1/4) \in [17, 64]$, and $N(1/8) \in [67, 512]$. (Theoretically, the lower bounds on $N(1/4)$ and $N(1/8)$ are 16 and 64, respectively, but since we scaled the pixel values to [–8, 8] and at least one pixel takes on a scaled value of an endpoint, this forces these lower bounds to be slightly higher for flat surfaces.) We may include a step in the algorithm to identify "completely" flat surfaces, to which we assign a local fractal dimension of 2.0. This method clearly places our computed local fractal dimension in the interval [2, 3]. (Naturally, for a computer implementation where speed is important, the division by 10 ln(2) is omitted in the interest of computing time.)

8. Implementation and Modifications

To make the final characterization of an object, we start with a bank of test images to which we apply our algorithm. We then create a histogram of the local fractal-dimension values within the objects of interest, from which we create a look-up table based on an appropriate range. We search our processed images for these values to detect possible objects of interest. Methods exist that significantly reduce the search time (Anderson, 1993), and these may also be applied to reduce the number of pixels for which we actually compute the fractal dimension; however, these are not critical to the presentation of our algorithm.

While our algorithm is tailored to a specific application, we have presented the procedure as generally as possible so that it might be tailored for other applications. This is best done through experimentation, the proof of which is the final result. In our case, we want the fractal dimension of a manmade object to be distinguishable from the desert sand (and perhaps even sensitive enough to distinguish among classes of vehicles); but the range of our look-up tables must allow us to achieve an appropriate balance between false positive and false negative readings. Since local fractal dimension is not unique, false positive detections are always possible, but too narrow a window in our look-up tables may lead to false negative readings, which are unacceptable for our application.

A 16×16 window size is appropriate for our application, based on the image source and a specific resolution, but this can certainly be modified as the source changes. For example, as the altitude of a passive image source increases, the objects of interest contain fewer pixels, and it may be appropriate to decrease the size of the local window. Additionally, since this local fractal dimension is not scale-invariant, we need to recalibrate our look-up tables. Clearly, for other applications, the optimal window size can be determined through experimentation.

We scale each window to the interval [–8, 8] so that the local fractal dimension is based entirely on the "shape" of the object, and we do not re-scale these computed values based on the range of original pixel values. For black-and-white photography, which is sensitive to different light conditions, this may not be appropriate. Instead, the computed values can be scaled by an appropriate factor based on the dynamic range of pixel values as mentioned above.

We can further refine our procedure to make it even more sensitive to slight differences in pixel values, although this may degrade the robustness of the algorithm for some purposes. Since strictly counting boxes produces the same result whether the greatest pixel value in our square is 6.1 or 8.0, we may increase the sensitivity by using fractions of boxes. For example, while a pixel value of 8 requires four boxes of side 2 to cover it, a pixel value of 6.1 requires only 3.05 boxes, so we can use that value in our algorithm. To ensure that our computed fractal dimension remains between two and three, however, we should use fractions of boxes only when more than one box is required to cover a set of pixels. This modification may be appropriate for applications that require increased sensitivity.

If we are concerned with actually assigning a value to each pixel in our image, we can take the average of the four surrounding computed dimensions as the value we assign, but for most applications where speed is important, this step is unnecessary.

A disadvantage of this procedure is that it leaves a band eight pixels wide around the border of the image. The algorithm can be modified for pixels near the border, but with an image such as an x-ray where the area of concern is generally in the center of the image, or in a series of aerial images with overlap between adjacent images, we need not worry about the border.

This procedure produces a higher fractal dimension near the edges of objects, and while our goal is not to create an edge detector, this feature may be useful in our attempt to identify objects. As the fractal dimension of natural features varies smoothly across an image (Super, 1990), the fact that this procedure has jumps near the edges of objects can be used to the advantage of the analyst, particularly when he or she is attempting to distinguish man-made objects from a natural background.

References

Anderson, P. G. (1993), *Linear Pixel Shuffling for Image Processing, an Introduction,* J. Electronic Imaging, April, 147–154.

Falconer, K. (1991), *Fractal Geometry,* John Wiley & Sons, New York.

Gage, C. (1990), *Khoros User's Manual,* University of New Mexico.

Hentschel, H.G.E., and I. Procaccia (1983), *The Infinite Number of Generalized Dimensions of Fractals and Strange Attractors,* Physica, North-Holland Publishing Company.

Keller, J. M., S. Chen, and R. M. Crownover (1989), *Texture Description and Segmentation through Fractal Geometry,* Comput. Vision Graphics Image Proc., **45**, 150–166.

Mandelbrot, B. B. (1983), *The Fractal Geometry of Nature,* W. H. Freeman, San Francisco.

Milman, V. Y, N. A. Stelmashenko, and R. Blumenfeld (1994), *Fracture Surfaces: A Critical Review of Fractal Studies and a Novel Morphological Analysis of Scanning Tunneling Microscopy Measurements,* Prog. Mater. Sci., **38**, 425–474.

Moghaddam, B. (1991), *Local Fractal Dimension Operators and Relaxation Techniques for Image Segmentation,* Master's thesis, George Mason University.

Novak, L.M., G. J. Owirka, and C. M. Netishen (1993), *Performance of a High-Resulution Polarimetric SAR Automatic Target Recognition System,* The Lincoln Laboratory Journal, **6**, 11–24.

Super, B. J., and A. C. Bovick (1990), "Optimally Localized Estimation of the Fractal Dimension," in *Curves and Surfaces in Computer Vision and Graphics,* SPIE, **1251**, 357–368.

Turcotte, D. L. (1992), *Fractals and Chaos in Geology and Geophysics,* Cambridge University Press, Great Britain.

Voss, R. F. (1986), "Random Fractals, Characterization and Measurement," in *Scaling Phenomena in Disordered Systems* (R. Pynn and A. Skjeltorp, Eds.), Plenum, New York.

Distribution

Admnstr
Defns Techl Info Ctr
Attn DTIC-DDA (2 copies)
Cameron Sta Bldg 5
Alexandria VA 22304-6145

Dept of Mathematics
US Military Academy
Attn COL C Arney
West Point NY 10996

HQ Dept of the Army
Attn DAMO-FDQ MAJ McGonagle
400 Army Pentagon
Washington DC 20310-0460

Dept of Mathematics
Nav Postgraduate School
Attn MAJ P Beaver (10 copies)
Monterry CA 93943

Dept of Electrl & Cmptr Engrg
State Univ of NY at Buffalo
Attn N Nasrabadi
Amherst NY 14260

Dept of Mathematics
Univ of Santa Clara
Attn J Leader
Santa Clara CA 95053

US Army Rsrch Lab
Attn AMSRL-SE-RS T Kipp
Attn AMSRL-SE-RT B Weber
FT Belvoir VA 22060

US Army Rsrch Lab
Attn AMSRL-SE-RT M Hamilton
Attn AMSRL-SE-RT V Mirelli
Bldg 357 Rm 231
FT Belvoir, VA 22060

US Army Rsrch Lab
Attn AMSRL-IS-TA B Sadler
Attn AMSRL-IS-TA D Torrieri
Attn AMSRL-IS-TA S Hayes
Attn AMSRL-OP-SD-TA Mail & Records Mgmt
Attn AMSRL-OP-SD-TL Tech Library (3 copies)
Attn AMSRL-OP-SD-TP Tech Pub
Attn AMSRL-SE J M Miller
Attn AMSRL-SE J Pellegrino
Attn AMSRL-SE J Sattler (20 copies)
Attn AMSRL-SE-EO J Mait
Attn AMSRL-SE-R E Burke
Attn AMSRL-SE-RI D Rodkey
Attn AMSRL-SE-RI G H Stolovy
Attn AMSRL-SE-RI J Dammann
Attn AMSRL-SE-RU J McCorkle
Attn AMSRL-SE-RU J Sichina
Attn AMSRL-SE-RU K Sturgess
Attn AMSRL-SE-RU R Kapoor
Attn AMSRL-SE-RU V Sabio
Attn AMSRL-SSE-EO A Filipov
Attn AMSRL-SE-RI A M P Marinelli
Attn AMSRL-SE-RI D S Rosario
Attn AMSRL-SL-CN C Glenn

DEPARTMENT OF THE ARMY
U.S. Army Research Laboratory
2800 Powder Mill Road
Adelphi, MD 20783-1197

An Equal Opportunity Employer